19장 프로토타입

자바스크립트는 클래스 기반 객체지형 프로그래밍 언어보다 효율적이며 더 강력한 객체지향 프로그래밍 능력을 지니고 있는 프로토타입 기반의 객체지향 프로그래밍 언어.

자바스크립트를 이루고 있는 거의 "모든 것"이 객체이다.

19.1 객체지향 프로그래밍

- 객체 지향 프로그래밍 : 절차지향적 관점에서 벗어나 여러 개의 독립적 단위(객체)의 집합으로 프로그램을 표현.
- 속성 : 실체가 가지는 특징이나 성질.
- 추상화 : 다양한 속성 중에서 프로그램에 필요한 속성만 간추려 내어 표현하는 것.
- 객체 : 속성을 통해 여러 개의 값을 하나의 단위로 구성한 복합적인 자료구조, 상태 데이터와 동작을 하나의 논리적인 단위로 묶은 복합적인 자료구조.

19.2 상속과 프로토타입

- 상속 : 어떤 객체의 프로퍼티 또는 메서드를 다른 객체가 상속받아 그대로 사용할 수 있는 것.
- 자바스크립트는 프로토타입을 기반으로 상속을 구현하여 불필요한 중복을 제거한다(기존 코드 재사용).

```
// 생성자 함수
function Circle(radius) {
    this.radius = radius;
    this.getArea = function () {
            // Math.PI는 원주율을 나타내는 상수다.
            return Math.PI * this.radius ** 2;
    };
}

// 반지름이 1인 인스턴스 생성
const circle1 = new Circle(1);
```

```
// 반지름이 2인 인스턴스 생성
const circle2 = new Circle(2);

// Circle 생성자 함수는 인스턴스를 생성할 때마다 동일한 동작을 하는
// getArea 메서드를 중복 생성하고 모든 인스턴스가 중복 소유한다.
// getArea 메서드는 하나만 생성하여 모든 인스턴스가 공유해서 사용하는
것이 바람직하다.
console.log(circle1.getArea === circle2.getArea);// false
console.log(circle1.getArea());// 3.141592653589793
console.log(circle2.getArea());// 12.566370614359172
```

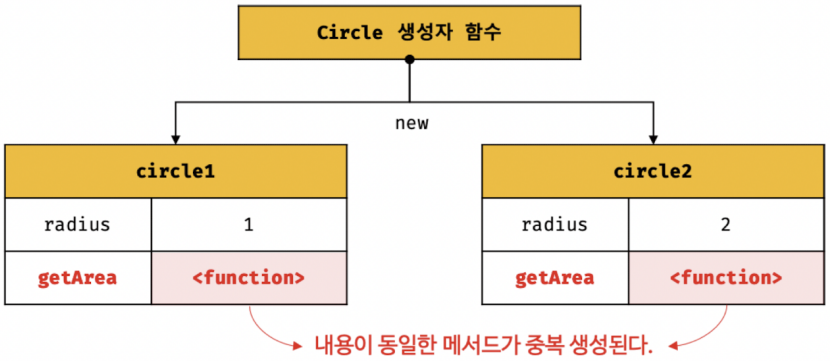

그림 19–1 메서드 중복 생성

https://velog.io/@jjinichoi/모던-자바스크립트-Deep-Dive-19장-프로토타입

```
// 생성자 함수
function Circle(radius) {
  this.radius = radius;
}

// Circle 생성자 함수가 생성한 모든 인스턴스가 getArea 메서드를
// 공유해서 사용할 수 있도록 프로토타입에 추가한다.
// 프로토타입은 Circle 생성자 함수의 prototype 프로퍼티에 바인딩되어
있다.
Circle.prototype.getArea = function () {
  return Math.PI * this.radius ** 2;
};
```

```
// 인스턴스 생성
const circle1 = new Circle(1);
const circle2 = new Circle(2);

// Circle 생성자 함수가 생성한 모든 인스턴스는 부모 객체의 역할을 하는
// 프로토타입 Circle.prototype으로부터 getArea 메서드를 상속받는다.
// 즉, Circle 생성자 함수가 생성하는 모든 인스턴스는 하나의 getArea
메서드를 공유한다.
console.log(circle1.getArea === circle2.getArea);// true
console.log(circle1.getArea());// 3.141592653589793
console.log(circle2.getArea());// 12.566370614359172
```

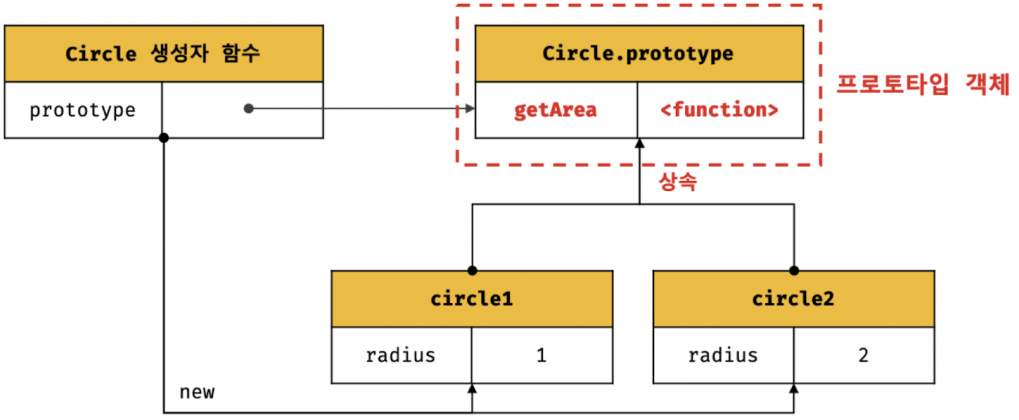

그림 19–2 상속에 의한 메서드 공유

https://velog.io/@jjinichoi/모던-자바스크립트-Deep-Dive-19장-프로토타입

19.3 프로토타입 객체

- 프로토타입 객체(=프로토타입)
  - 객체 간 상속을 구현하기 위해 사용
  - 어떤 객체의 상위(부모) 객체의 역할을 하는 객체로서 다른 객체에 공유 프로퍼티(메서드 포함)를 제공한다.
  - 프로토타입을 상속받은 하위(자식) 객체는 상위 객체의 프로퍼티를 자신의 프로퍼티처럼 자유롭게 사용 가능.

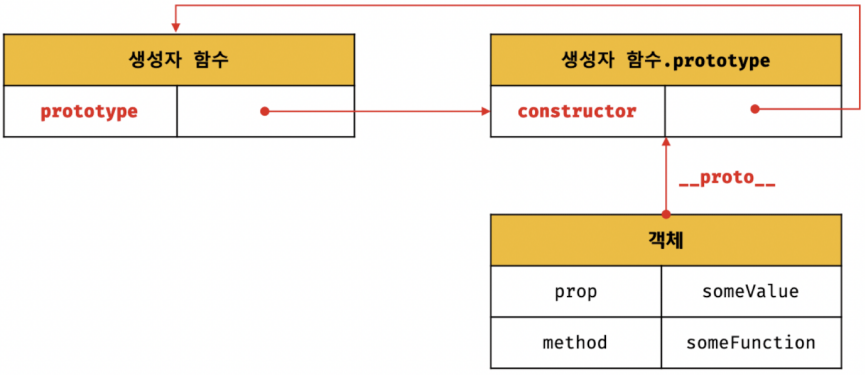

그림 19–3 객체와 프로토타입과 생성자 함수는 서로 연결되어 있다.

19.3.1 __proto__ 접근자 프로퍼티

- 모든 객체는 __proto__ 접근자 프로퍼티를 통해 자신의 프로토타입, 즉 [[Prototype]] 내부 슬롯에 간접적으로 접근 가능.
- __proto__ 접근자 프로퍼티를 코드 내에서 직접 사용 권장X (직접 상속 등의 경우에 사용 불가)
    - → Object.getPrototypeOf 메서드로 프로토타입의 참조 취득,
      Object.setPrototypeOf 메서드로 프로토타입 교체.

19.3.2 함수 객체의 prototype 프로퍼티

- prototype 프로퍼티 : 함수 객체만이 소유함(일반 객체는 X),
  생성자 함수가 생성할 인스턴스의 프로토타입을 가리킨다.

19.3.3 프로토타입의 constructor 프로퍼티와 생성자 함수

- constructor 프로퍼티
    - 모든 타입은 constructor 프로퍼티를 가진다.
    - prototype 프로퍼티로 자신을 참조하고 있는 생성자 함수를 가리킨다.
    - 생성자 함수가 생성될 때(함수 객체 생성될 때) 연결된다.

19.4 리터럴 표기법에 의해 생성된 객체의 생성자 함수와 프로토타입

- 프로토타입과 생성자 함수는 언제나 쌍으로 존재한다.
- 생성자 함수에 의해 생성된 인스턴스는 프로토타입의 constructor 프로퍼티에 의해 생성자 함수와 연결된다.

19.5 프로토타입의 생성 시점

- 프로토타입은 생성자 함수가 생성되는 시점에 더불어 생성

19.5.1 사용자 정의 생성자 함수와 프로토타입 생성 시점

- 생성자 함수로서 호출할 수 있는 함수, 즉 constructor는 함수 정의가 평가되어 함수 객체를 생성하는 시점에 프로토타입도 더불어 생성
- 생성자 함수로서 호출할 수 없는 함수, 즉 non-constructor는 프로토타입이 생성되지 않는다.(ex 화살표 함수)

19.7 프로토타입 체인

- 프로토타입 체인 : 자바스크립트는 객체의 프로퍼티(메서드 포함)에 접근하려고 할 때 해당 객체에 접근하려는 프로퍼티가 없다면 [[Prototype]] 내부 슬롯의 참조를 따라 자신의 부모 역할을 하는 프로토타입의 프로퍼티를 순차적으로 검색한다.(상속과 프로퍼티 검색을 위한 메커니즘)
- Object.prototype : 프로토타입 체인의 종점

  모든 객체가 상속받음.

19.8 오버라이딩과 프로퍼티 섀도잉

- 프로퍼티 섀도잉 : 상속 관계에 의해 프로퍼티가 가려지는 현상.

19.10 instanceof 연산자

- 객체 instanceof 생성자 함수 : 우변의 생성자 함수의 prototype에 바인딩된 객체가 좌변의 객체의 프로토타입 체인 상에 존재하면 true로 평가되고, 그렇지 않은 경우에는 false로 평가

19.11 직접 상속

19.11.1 Object.create에 의한 직접 상속

- Object.create 메서드 : 명시적으로 프로토타입을 지정하여 새로운 객체 생성
  - 첫번째 매개변수에는 생성할 객체의 프로토타입으로 지정할 객체를 전달
  - 두번째 매개변수에는 생성할 객체의 프로퍼티 키와 프로퍼티 디스크립터 객체로 이뤄진 객체 전달(생략 가능)

19.12 정적 프로퍼티/메서드

- 정적 프로퍼티/메서드 : 생성자 함수로 인스턴스를 생성하지 않아도 참조/호출할 수 있는 프로퍼티/메서드

19.13 프로퍼티 존재 확인

19.13.1 in 연산자

- in 연산자 : 객체 내에 특정 프로퍼티가 존재하는지 여부를 확인.
- key in object : key(프로퍼티 키를 나타내는 문자열), object(객체로 평가되는 표현식)

19.13.2 Object.prototype.hasOwnProperty 메서드

- Object.prototype.hasOwnProperty 메서드 : 인수로 전달받은 프로퍼티 키가 객체 고유의 프로퍼티 키인 경우에만 true를 반환하고 상속받은 프로토타입의 프로퍼티 키인 경우 false를 반환

19.14 프로퍼티 열거

19.14.1 for...in 문

- for...in 문
  - 객체의 모든 프로퍼티를 순회하며 열거
  - 객체의 프로토타입 체인 상에 존재하는 모든 프로토타입의 프로퍼티 중에서 프로퍼티 어트리뷰트
  - [[Enumerable]]의 값이 true인 프로퍼티를 순회하며 열거

```
const person = {
  name: 'Lee',
  address: 'Seoul',
  __proto__: { age: 20 }
};

for (const key in person) {
  console.log(key + ': ' + person[key]);
}
// name: Lee
// address: Seoul
// age: 20
```

19.14.2 Object.keys/values/entries 메서드

- 객체 자신의 고유 프로퍼티만 열거